I. Data Preprocessing

Pandas will load the dataset. It could find there are 367230×12 data in this dataset after statistics. Also, there is missing value in ContractType, ContractTime and Company which shows in TableI and TableII.

TABLE I
THE STATISTICS OF THE ORIGINAL TRAINNING DATASET

	Id	244767 non-null int64
	Title	244767 non-null object
	FullDescription	244767 non-null object
	LocationRaw	244767 non-null object
	LocationNormalized	244767 non-null object
Training data	ContractType	65442 non-null object
	ContractTime	180863 non-null object
	Company	212338 non-null object
	Category	244767 non-null object
	SalaryRaw	244767 non-null object
	SalaryNormalized	244767 non-null object
	SourceName	244767 non-null object

TABLE II
THE STATISTICS OF THE ORIGINAL TESTING DATASET

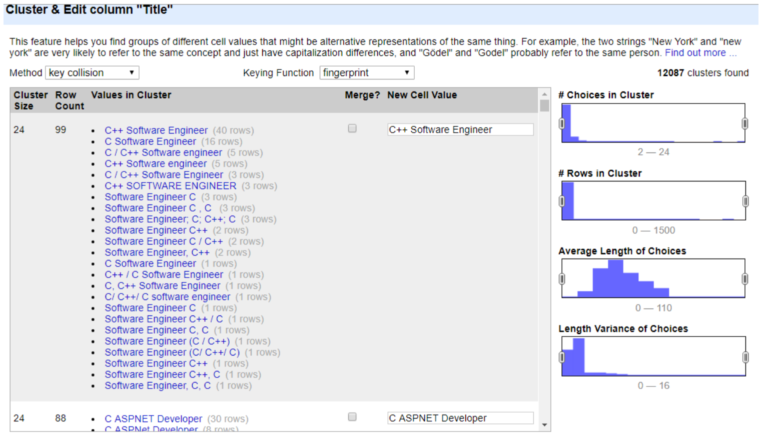
	Id	122463 non-null int64
	Title	122463 non-null object
	FullDescription	122463 non-null object
	LocationRaw	122463 non-null object
	LocationNormalized	122463 non-null object
Testing data	ContractType	33013 non-null object
	ContractTime	90702 non-null object
	Company	106202 non-null object
	Category	122463 non-null object
	SalaryRaw	NA
	SalaryNormalized	NA
	SourceName	122463 non-null object

- Clustering

  Openrefine is a data transforming tool which could search, clean and integrate data[1]. In this project, the missing value will be filled by this tool and the string data will be clustered which is shown in Fig1.
- Location match

  Location tree is a supplemental dataset which contains the hierarchical relationship between the different Normalized Locations of the original dataset. Therefore, the longest word in each data of LocaitonNormalized will be used to match the information in location tree.
- Coding and One-hot-vector

  For the information in FullDescription, the HasingVectrizer of sklearn will be used to process string information with Hash coding. Each cluster of attributes will be coded with a unique digital. Due to there are too many clusters in Title, Location and Company, only the number of cluster bigger than ten will be counted as valid clusters. After coding, the one-hot encoding module in Keras will be used to generate the one-hot vector of unique digital.



Cluster & Edit column "Title"

This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more ...

Method: key collision

Keying Function: fingerprint

12087 clusters found

Cluster Size	Row Count	Values in Cluster	Merge?	New Cell Value
24	99	C++ Software Engineer (40 rows) C Software Engineer (16 rows) C / C++ Software engineer (5 rows) C++ Software engineer (5 rows) C / C++ Software Engineer (3 rows) C++ SOFTWARE ENGINEER (3 rows) Software Engineer C (3 rows) Software Engineer C , C (3 rows) Software Engineer; C; C++; C (3 rows) Software Engineer C++ (2 rows) Software Engineer C / C++ (2 rows) Software Engineer, C++ (2 rows) C Software Engineer (1 rows) C++ / C Software Engineer (1 rows) C, C++ Software Engineer (1 rows) C/ C++/ C software engineer (1 rows) Software Engineer C (1 rows) Software Engineer C++ / C (1 rows) Software Engineer C, C (1 rows) Software Engineer (C / C++) (1 rows) Software Engineer (C/ C++/ C) (1 rows) Software Engineer C++ (1 rows) Software Engineer C++, C (1 rows) Software Engineer, C, C (1 rows)	□	C++ Software Engineer
24	88	C ASPNET Developer (30 rows) C ASPNet Developer (8 rows)	□	C ASPNET Developer

Fig. 1. Clustering function in Openrefine

After data preprocessing, the orgiAfter data preprocessing, the original dataset is transforming to digital dataset with shape of 367230×9736.

II. KNN(K-nearest neighbors algorithm)

When training dataset with given tags, the feature of test data can be compared with features in training data. Then, the top K of most similar features in training dataset will be calculated and the test data will be classified to the most frequently occurring tags[2]. The algorithm is described as:

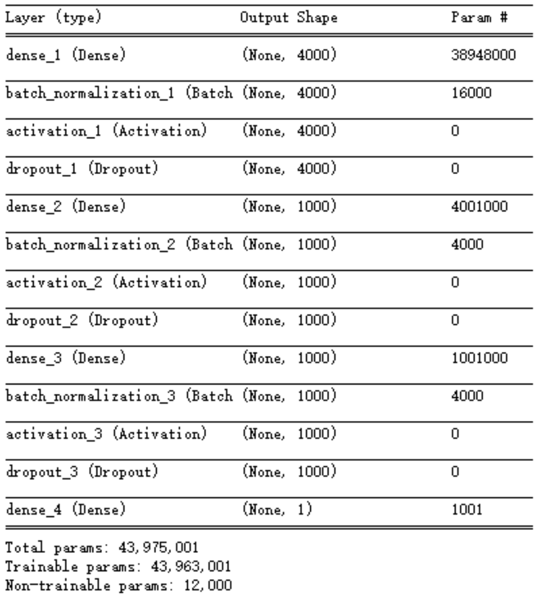
1 Calculate the distance between test data and each training data.
2 Sort by the relationship of increasing distance.
3 Select the K points which have the smallest distance.
4 Determine the occurred frequency of the top K points.
5 Returns the most frequent category of the top K points as a predictive classification.

The training data will be divided into ten groups by KFold and the Grid Search in sklearn will be used to find the best value of K in KNN. In this project, the best parameter of K is 1. Therefore, training all data with one neighbor and the mean difference of predicting salary is 15850.69. This result is unsatisfactory because the dimensional of data is high and KNN will give an existing salary(tag) in training data which will also cause the error.

III. NN(Artificial neural network)

Several models were built by Keras in this part. Due to the page limitation, the model with the best performance will be introduced. Fig2 shows the configuration of this model.

- It uses Sequential model for building and the dense is full connected layers which could represent different weights.
- Batch normalization is using to adjust the parameters between layers by zero mean value and one variance[3]. In this project, the convergence speed increased striking with BN preprocessing.
- Rectified Linear Unit(ReLU) is used as activating the function to add a non-linear parameter to dense. It has six times convergence speed compared to sigmoid and tanh function and it could reduce the vanishing gradient[4].
- Dropout means the connection between each neural has a disconnected probability which could reduce the complexity of the model to prevent overfitting[5]. Also, and the call back function is used to avoid overfitting with early stopping as well.



```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 4000)              38948000
_________________________________________________________________
batch_normalization_1 (Batch (None, 4000)              16000
_________________________________________________________________
activation_1 (Activation)    (None, 4000)              0
_________________________________________________________________
dropout_1 (Dropout)          (None, 4000)              0
_________________________________________________________________
dense_2 (Dense)              (None, 1000)              4001000
_________________________________________________________________
batch_normalization_2 (Batch (None, 1000)              4000
_________________________________________________________________
activation_2 (Activation)    (None, 1000)              0
_________________________________________________________________
dropout_2 (Dropout)          (None, 1000)              0
_________________________________________________________________
dense_3 (Dense)              (None, 1000)              1001000
_________________________________________________________________
batch_normalization_3 (Batch (None, 1000)              4000
_________________________________________________________________
activation_3 (Activation)    (None, 1000)              0
_________________________________________________________________
dropout_3 (Dropout)          (None, 1000)              0
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 1001
=================================================================
Total params: 43,975,001
Trainable params: 43,963,001
Non-trainable params: 12,000
```

Fig. 2. Structure of training model

Three optimizers were tested in this project which are sgd(mini-batch gradient descent), Adadelta and Adam(Adaptive Moment Estimation). Adam had a higher convergence speed, but sgd had a higher accuracy.

Therefore, the final configuration is a three layers model with sgd optimizer. After epochs 2000, the changing in mean squared error of model is shown in Fig3. It could find the convergence speed is decreased with epochs increasing, this is because the learning rate is a constant in this project. The MSE of this model still decreases, however, due to the time limitation and the slow convergence speed. The training stops at epochs 2000. Comparing with KNN, the predicted salary of this NN model has outstanding progress. The mean difference of this model is 9275.03 which could rank 77 of this competition. In addition, the author of this competition did not provide exact salary information of test dataset but give a predicted result with the Random solemnity which means the mean difference may reduce with real data.

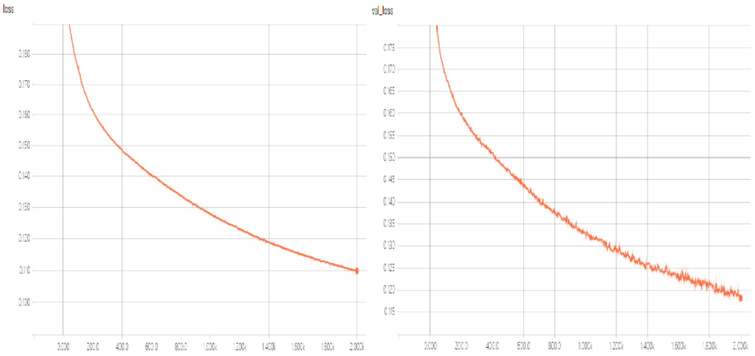


Fig. 3. MSE changing of training data and valid data in epoch 2000

References

[1] R. Verborgh and M. De Wilde, *Using OpenRefine*. Packt Publishing Ltd, 2013.

[2] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[4] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[5] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.